Week 12 Report

Thomas Deatherage
NFHM/BioCosmos
8 November 2024

Time Slot

1) What progress did you make in the last week?

- Weekly catchup with Roman and Bree
- Performance Evaluation of VLM4Bio Unicom-trained CLIP model
- Fixed gaps in captioning dataset

2) What are you planning on working on next?

- Comparing a base untrained CLIP architecture vs a pre-trained CLIP architecture via the Unicom technique.
- Check-in with Dr Porto and Dr Moritz to see their thoughts are

3) Is anything blocking you from getting work done?

- Nope.

Abstracts & Summaries

Towards a Taxonomy Machine – A Training Set of 5.6 Million Arthropod Images

Abstract: The taxonomic identification of organisms from images is an active research area within the machine learning community. Current algorithms are very effective for object recognition and discrimination, but they require extensive training datasets to generate reliable assignments. This study releases 5.6 million images with representatives from 10 arthropod classes and 26 insect orders. All images were taken using a Keyence VHX-7000 Digital Microscope system with an automatic stage to permit high-resolution (4K) microphotography. Providing phenotypic data for 324,000 species derived from 48 countries, this release represents, by far, the largest dataset of standardized arthropod images. As such, this dataset is well suited for testing the efficacy of machine learning algorithms for identifying specimens into higher taxonomic categories.

Summary: This document details the creation of a massive dataset containing 5.6 million high-resolution images of arthropods, predominantly insects, for use in machine learning-based taxonomy and biodiversity research. Captured using the Keyence VHX-7000 microscope, these images represent over 320,000 species from 48 countries and aim to support the development of automated image-based classification systems for arthropod identification. This dataset, one of the largest and most geographically diverse of its kind, is positioned as a vital resource for advancing machine learning applications in taxonomy
Relevance: Dr Porto shared this paper with us. Such a dataset is highly relevant to our task of creating a high-performance species search database. We need data.
https://www.mdpi.com/2306-5729/9/11/122

Scripts and Code Blocks

This week I focused on performance evaluation. Specifically, I benchmarked recall performance on the trained model, comparing generated test embeddings against their most similar neighbors and checking category, genus and species. The most important part of that code is reproduced below:

```
for images, cat_labels, sp_labels, categories, species in test_loader:
    images = images.cuda()
    embeddings = model(images)
    embeddings = F.normalize(embeddings, p=2, dim=1)

    all_embeddings.append(embeddings.cpu())
    all_category_labels.append(cat_labels)
    all_species_labels.append(sp_labels)
    all_categories.extend(categories)
    all_species.extend(species)

# Concatenate all embeddings and labels
all_embeddings = torch.cat(all_embeddings, dim=0)
all_category_labels = torch.cat(all_category_labels, dim=0)
all_species_labels = torch.cat(all_species_labels, dim=0)

# Compute similarity matrix
similarity = torch.mm(all_embeddings, all_embeddings.t())

# Remove diagonal elements
```

```
    mask = torch.eye(similarity.shape[0], dtype=bool)
    similarity[mask] = -float('inf')

    # Compute metrics for different k values
    ks = [1, 5, 10]
    metrics = {}

    for k in ks:
        _, indices = similarity.topk(k, dim=1)

        # Per-category metrics
        category_metrics = {}
        for category in set(all_categories):
            category_mask = [c == category for c in all_categories]
            category_indices = [i for i, m in enumerate(category_mask) if
m]

            if not category_indices:
                continue

            correct = 0
            for idx in category_indices:
                retrieved_categories = [all_categories[i] for i in
indices[idx]]
                if category in retrieved_categories:
                    correct += 1

            recall = correct / len(category_indices)
            category_metrics[category] = {
                'recall': recall,
                'count': len(category_indices)
            }

        # Per-species metrics
        species_metrics = {}
        for species in set(all_species):
            species_mask = [s == species for s in all_species]
            species_indices = [i for i, m in enumerate(species_mask) if m]

            if not species_indices:
                continue

            correct = 0
            for idx in species_indices:
                retrieved_species = [all_species[i] for i in indices[idx]]
                if species in retrieved_species:
                    correct += 1

            recall = correct / len(species_indices)
            species_metrics[species] = {
                'recall': recall,
                'count': len(species_indices),
                'category': all_categories[species_indices[0]]  # Store
category for grouping
            }

        # Overall metrics
        category_correct = 0
        species_correct = 0
        total = len(all_category_labels)

        for i, (cat_label, sp_label) in enumerate(zip(all_category_labels,
all_species_labels)):
            retrieved_cat_labels = all_category_labels[indices[i]]
            retrieved_sp_labels = all_species_labels[indices[i]]

            if cat_label in retrieved_cat_labels:
                category_correct += 1
            if sp_label in retrieved_sp_labels:
                species_correct += 1

        metrics[f'recall@{k}'] = {
            'overall_category': category_correct / total,
            'overall_species': species_correct / total,
            'per_category': category_metrics,
            'per_species': species_metrics
        }

    return metrics
```

Otherwise, all the code provided by Unicom basically remained the same.

Flow Charts/Diagrams

Nothing new to show here

Documentation

No documentation to add really just yet

Results Visualization + Proof of Work

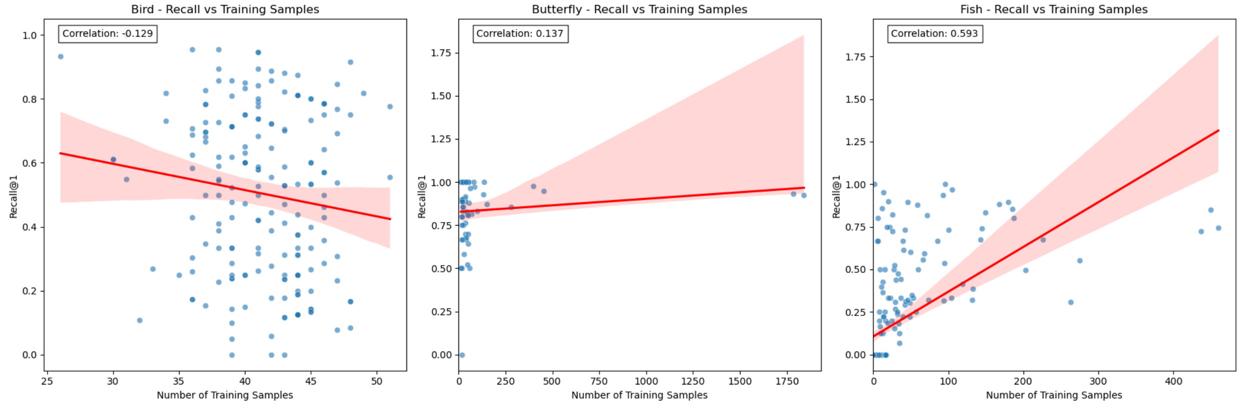
Overall Improvements, base (ViT/B-32 CLIP) vs trained CLIP

Trained
Overall Category Recall: 0.9995
Overall Species Recall: 0.**6508**

Base
Overall Category Recall: 0.9996
Overall Species Recall: 0.**5784**

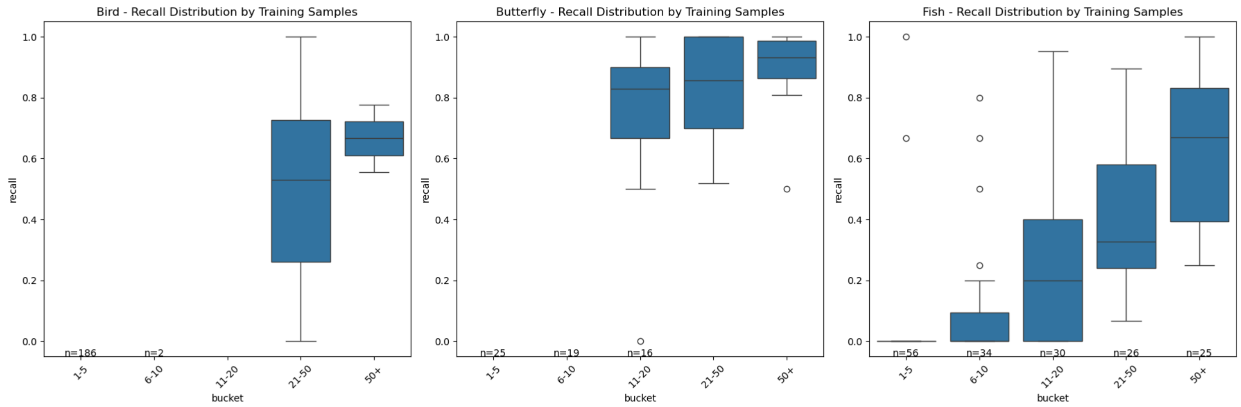
So a **.724** point improvement overall in species recall in the trained model vs the base model. Category (Fish, Bird, or Butterfly) recall was near perfect in both cases, which wasn't surprising.

However, recall performance varied by species. And, in the case of Fish and Butterfly, species with more training samples exhibited better recall performance:



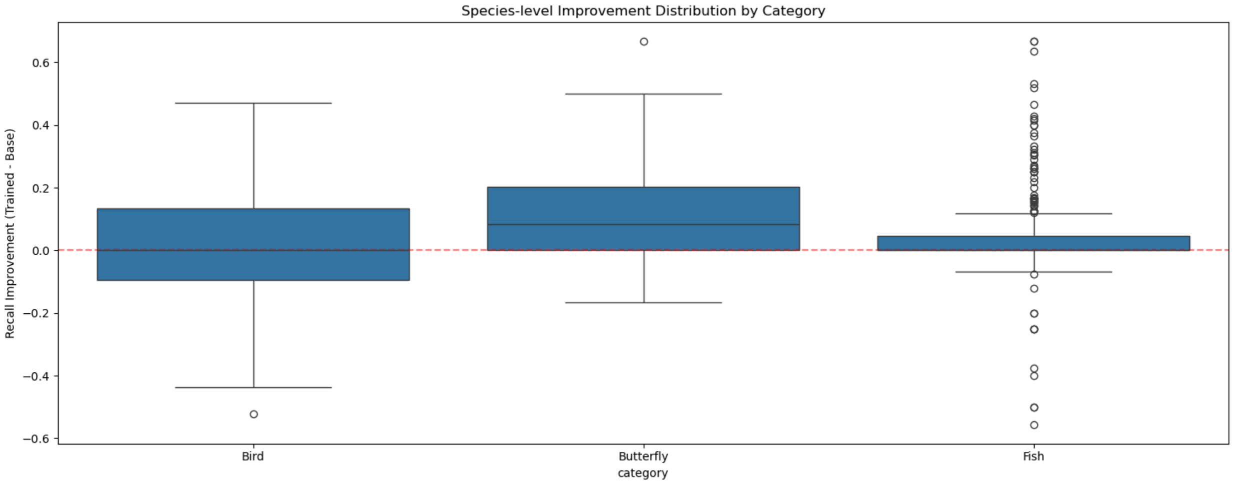
Bird's a little bit of an anomaly here, as its range of species training sample sizes clustered in the 25-51 range. I don't think that negative correlation is super meaningful.

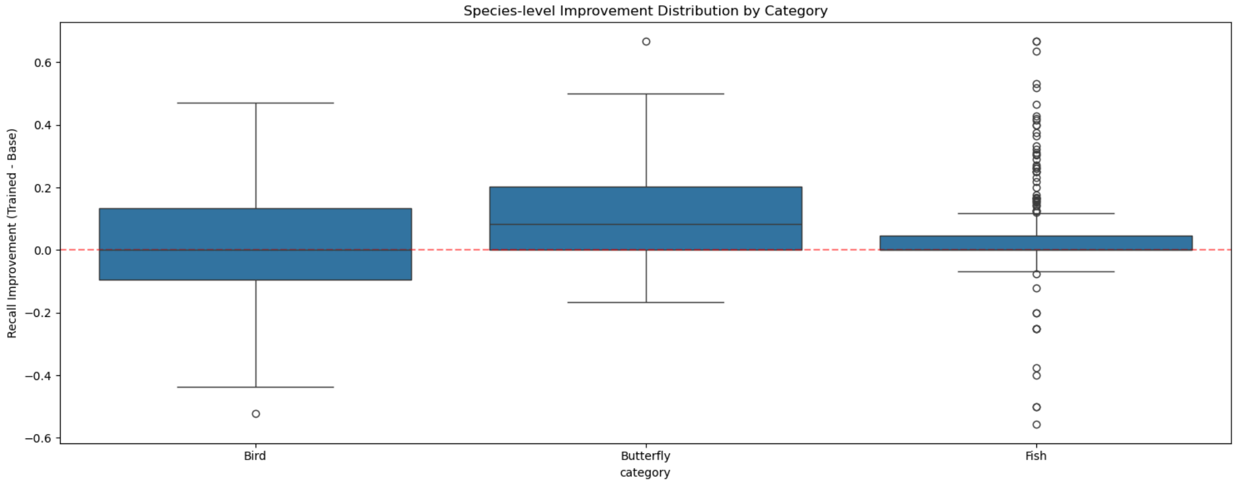
The above data presented as box plots shows the same general idea:



The next interesting thing was comparing the trained model's recall performance vs the base ViT/B-32 model:

Butterfly was the only category that seemed to exhibit unmistakable median improvement over the base model by both species and genus recall:





The fact that we see such variation in species improvement is interesting. And of course the trained model actually performed worse for some species. My hunch is that training sample size had an effect, which I'll look into next week. Fortunately, as the box plots kind of suggest, whiskers and IQR seems to tend *up* toward general improvement, rather than worsening performance.

Some more extensive details about which species the model did well with . . .and which it didn't are provided below. I'll be visually inspecting these results more closely next week.

Comparison Analysis of Base vs Trained Model

```
==========================================

Bird:
-----

Species-level Analysis:
Number of species: 188

Recall Statistics:
Base Model - Mean: 0.501, Median: 0.500
Trained Model - Mean: 0.504, Median: 0.533
Average Improvement: 0.003

Top 5 Most Improved Species:
larus occidentalis: 0.471 (Base: 0.235 → Trained: 0.706)
phoebastria nigripes: 0.348 (Base: 0.478 → Trained: 0.826)
```

```
helmitheros vermivorum: 0.316 (Base: 0.316 → Trained: 0.632)
aethia cristatella: 0.308 (Base: 0.692 → Trained: 1.000)
sterna paradisaea: 0.286 (Base: 0.381 → Trained: 0.667)

Top 5 Least Improved Species:
ammospiza leconteii: -0.522 (Base: 0.696 → Trained: 0.174)
anthus rubescens: -0.438 (Base: 0.562 → Trained: 0.125)
campylorhynchus brunneicapillus: -0.389 (Base: 0.667 → Trained: 0.278)
parkesia noveboracensis: -0.333 (Base: 0.417 → Trained: 0.083)
aethia psittacula: -0.333 (Base: 0.833 → Trained: 0.500)

Genus-level Analysis:
Number of genera: 114

Recall Statistics:
Base Model - Mean: 0.554, Median: 0.571
Trained Model - Mean: 0.547, Median: 0.559
Average Improvement: -0.008

Top 5 Most Improved Genera:
helmitheros: 0.316 (Base: 0.316 → Trained: 0.632)
pipilo: 0.272 (Base: 0.372 → Trained: 0.644)
sterna: 0.242 (Base: 0.281 → Trained: 0.523)
passerella: 0.222 (Base: 0.500 → Trained: 0.722)
fulmarus: 0.217 (Base: 0.478 → Trained: 0.696)

==================================================
```

Butterfly:

```
----------

Species-level Analysis:
Number of species: 60

Recall Statistics:
Base Model - Mean: 0.714, Median: 0.762
Trained Model - Mean: 0.836, Median: 0.889
Average Improvement: 0.122

Top 5 Most Improved Species:
```

```
pyrrhogyra cramen: 0.667 (Base: 0.333 → Trained: 1.000)
eueides isabella: 0.500 (Base: 0.000 → Trained: 0.500)
ithomia salapia: 0.429 (Base: 0.571 → Trained: 1.000)
catoblepia soranus: 0.400 (Base: 0.600 → Trained: 1.000)
zaretis isidora: 0.400 (Base: 0.400 → Trained: 0.800)

Top 5 Least Improved Species:
heliconius aoede: -0.167 (Base: 0.833 → Trained: 0.667)
lasaia agesilas: -0.143 (Base: 1.000 → Trained: 0.857)
heliconius demeter: -0.143 (Base: 0.143 → Trained: 0.000)
heliconius hecale: -0.133 (Base: 0.800 → Trained: 0.667)
taygetis thamyra: -0.120 (Base: 0.640 → Trained: 0.520)

Genus-level Analysis:
Number of genera: 27

Recall Statistics:
Base Model - Mean: 0.765, Median: 0.778
Trained Model - Mean: 0.884, Median: 0.900
Average Improvement: 0.119

Top 5 Most Improved Genera:
catoblepia: 0.400 (Base: 0.600 → Trained: 1.000)
prepona: 0.400 (Base: 0.500 → Trained: 0.900)
zaretis: 0.400 (Base: 0.400 → Trained: 0.800)
bia: 0.364 (Base: 0.545 → Trained: 0.909)
pyrrhogyra: 0.292 (Base: 0.483 → Trained: 0.775)

==================================================
```

Fish:

```
-----

Species-level Analysis:
Number of species: 249

Recall Statistics:
Base Model - Mean: 0.136, Median: 0.000
Trained Model - Mean: 0.179, Median: 0.000
Average Improvement: 0.043
```

```
Top 5 Most Improved Species:
brachyrhaphis parismina: 0.667 (Base: 0.000 → Trained: 0.667)
esox masquinongy: 0.667 (Base: 0.000 → Trained: 0.667)
carassius auratus: 0.636 (Base: 0.182 → Trained: 0.818)
notropis leuciodus: 0.533 (Base: 0.200 → Trained: 0.733)
notropis heterodon: 0.520 (Base: 0.360 → Trained: 0.880)

Top 5 Least Improved Species:
lepomis marginatus: -0.556 (Base: 0.556 → Trained: 0.000)
benthobatis yangi: -0.500 (Base: 0.500 → Trained: 0.000)
gambusia: -0.500 (Base: 0.500 → Trained: 0.000)
notropis lutipinnis: -0.400 (Base: 0.600 → Trained: 0.200)
lepomis microlophus: -0.375 (Base: 0.375 → Trained: 0.000)

Genus-level Analysis:
Number of genera: 86

Recall Statistics:
Base Model - Mean: 0.085, Median: 0.000
Trained Model - Mean: 0.120, Median: 0.000
Average Improvement: 0.034

Top 5 Most Improved Genera:
brachyrhaphis: 0.667 (Base: 0.000 → Trained: 0.667)
carassius: 0.636 (Base: 0.182 → Trained: 0.818)
priapella: 0.400 (Base: 0.400 → Trained: 0.800)
phoxinus: 0.322 (Base: 0.044 → Trained: 0.367)
opsopoeodus: 0.250 (Base: 0.000 → Trained: 0.250)
```

Next Week's Proposal

- Look into pretrained vs bare-bones model as base model
- Manual examination of species performance
- Check in with Dr Porto and Dr Moritz to show results